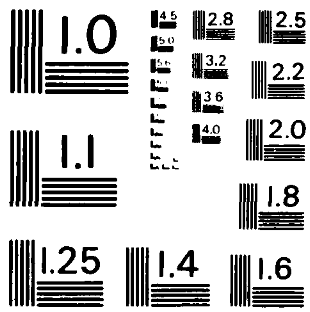
AD-A135 726 A STATISTICAL STUDY OF HARDWARE RELATED SOFTWARE ERRORS 1/1
IN MVS(U) STANFORD UNIV CA CENTER FOR RELIABLE
COMPUTING R K IYER ET AL. OCT 83 CRC-TR-83-12
UNCLASSIFIED ARO-18690.8-EL DAAG29-82-K-0105 F/G 9/2 NL

END
DATE
FILMED
-84
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

1

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ARO 18690.8-EL 12

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: CRC Tech. Rpt. 83-12	2. GOVT ACCESSION NO.: A135726
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): A statistical Study of Hardware Related Software Errors in MVS	5. TYPE OF REPORT & PERIOD COVERED: Interim Tech. Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Ravishankar K. Iyer and Paola Velardi	8. CONTRACT OR GRANT NUMBER(s): DAAG-29-82-K-0105
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Center for Reliable Computing, Computer Systems Laboratory, Stanford University; Stanford, CA 94305	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: DD Form 2222, Project No. P-18690-EL
11. CONTROLLING OFFICE NAME AND ADDRESS: U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709	12. REPORT DATE: October 1983
	13. NUMBER OF PAGES: 40
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Mr. James Gault; Electronics Division, U.S. Army Research Office, P.O. Box 12211, Research Triangle Park, NC 27709	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

A135726

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software reliability, hardware/software interactions, recovery

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper describes an analysis of hardware related software errors on the MVS operating system at the Center for Information Technology(CIT) at Stanford University. The study first examines the software error detection mechanisms with particular raference to the dection of software errors related to temporary and permanent hardware problems. About 11% of all software errors and over 40% of all software failures were found to be hardware related. It is shown that the system is seldom able to diagnose the fact that a software error may be

DD FORM 1473 (1 JAN 73) EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

83 12 13 022

/

hardware related. Key patterns in the occurence of hardware related software errors are determined and their effects on the system recovery examined. In a HW/SW record, both the hardware and the software errors occur in large clusters and have a significant percentage of lost records associated with them. The system recovery management is less likely to recover from hardware related software errors than software errors in general. It is suggested that error patterns found in this study could form the basis for the detection and recovery management of hardware related software errors.

Center for Reliable Computing

A Statistical Study of Hardware Related Software Errors in MVS

Ravishankar K. Iyer and Paola Velardi

CRC Technical Report No. 83-12

(CSL TN No. 83-231)

October 1983

CENTER FOR RELIABLE COMPUTING
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305

This work was supported in part by the Department of the Army under Contract Number DAAG29-82-K-0105, the Ugo Bordoni Foundation and the Italian National Research Council (CNR).

A Statistical Study of Hardware Related Software Errors in MVS

Ravishankar K. Iyer and Paola Velardi

CRC Technical Report No. 83-12

(CSL TN No. 83-231)

October 1983

CENTER FOR RELIABLE COMPUTING
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305

ABSTRACT

This paper describes an analysis of hardware related software errors on the MVS operating system at the Center for Information Technology (CIT) at Stanford University. The study first examines the software error detection mechanisms with particular reference to the detection of software errors related to temporary and permanent hardware problems. About 11 percent of all software errors and over 40 percent of all software failures were found to be hardware related. It is shown that the system is seldom able to diagnose the fact that a software error may be hardware related. Key patterns in the occurrence of hardware related software errors are determined and their effect on system recovery examined. In a HW/SW record, both the hardware and the software errors occur in large clusters and have a significant percentage of lost records associated with them. The system recovery management is less likely to recover from hardware related software errors than software errors in general. It is suggested that the error patterns found in this study could form the basis for the detection and recovery management of hardware related software errors.

Keywords: Software reliability, hardware/software interactions, recovery

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

The design of reliable and fault tolerant software systems is one of the most important issues facing computer designers today. Software cost and reliability are the major problem areas affecting modern computer systems. The question of hardware and software interaction and its effect on system reliability is particularly difficult to comprehend. It is further compounded by the lack of availability of real data. It is our view that results based on actual measurements and experiments are essential for developing a clear understanding of the problem.

The MVS system on the on the IBM 3081 at the Center for Information Technology (CIT) at Stanford University, provided an ideal opportunity in this regard. The operating system automatically collects information on error detection and correction. The state of the machine at the time of the error is also recorded. CIT is the main campus computation facility. It is used for production programs (payrolls and administration), student and research projects, and for general purpose computing. The installation consists of two IBM 3081 processors which run the MVS operating system. The two processors are loosely coupled, e.g., they have distinct control programs and different I/O configurations. On a typical day, the two systems support around 500 users and run approximately 4000 batch jobs.

The general objective of this study was to determine the extent and impact of temporary and permanent hardware errors on the operating system. The analysis differentiates between the terms "error" and "failure". A failure is an "error" which causes the termination of the system (i.e., a system failure). Thus an error, in general, may or may not

result in a failure. It is generally believed that the operating system is not always able to diagnose a software error related to a hardware error or failure. We define this as a hardware related software error and denote it as a "HW/SW" error. Note that the relationship may be either cause and effect (i.e. the hardware error caused the software error) or symptomatic (i.e both the hardware error and the software error are symptoms of another, yet unidentified, problem). A HW/SW error is further subdivided as follows:

1. Software errors found related to temporary hardware errors (denoted by "HW/SW-Temp.").
2. Software failures found related to permanent hardware failures (denoted by "HW/SW-Perm.").

We commence by analysing the error detection facilities in MVS with particular reference to hardware related problems. The most common types of hardware-related software errors are identified and their relative frequencies found. Finally the impact of HW/SW errors on the system is evaluated by measuring the effectiveness of system recovery in handling hardware related software errors.

The approach adopted was to start with a substantial quantity of high quality data on all software errors (recoverable and non-recoverable). The data on error detection and recovery is automatically logged by the operating system. An error collection mechanism which selected and filtered the raw data so as to cluster records referring to the same error, was developed (see [Velardi 83] for details). The data set so obtained was then merged with data sets of temporary and permanent hardware errors. The data on temporary hardware errors came from channel and

disk error logs [IBM 79]. Data on permanent hardware problems came from UNILOG [Butner 80], an installation (CIT) maintained log of failures and repair.

The important results of the study are summarised below:

1. The operating system is seldom able to diagnose the fact that a software error may be hardware related.
2. About 11 percent of all software errors were determined to be hardware related.
3. Over 40 percent of all software failures were found to be hardware related.
4. The key pattern of a HW/SW record is that both the hardware and the software records occur in large clusters and have a significant percentage of lost records associated with them.
5. The system recovery management is less effective in handling hardware related software errors than software errors in general.

Before describing this work in detail, an overview of related research in this area is presented.

2. RELATED RESEARCH AND MOTIVATION

Designing hardware systems that tolerate faults is relatively well understood, at least from a theoretical viewpoint. However, the problem of software fault tolerance has yet to be well investigated [Hecht 80a,b].

The term "software reliability model" is usually taken to mean mathematical relationships for assessing the reliability of software (in terms of statistical parameters such as Mean Time Between Failures) dur-

ing the development, debugging or testing phases. A few of these models have also been applied in follow-up operational phases. Several competing models have appeared in the literature (see [Musa 1980] for details), and a number of authors have attempted to analyse their suitability. An appreciation of the extent and nature of this discussion can be obtained from [Goel 80]. The main difficulty with these approaches is that, although each model appears to be valid within its own assumptions, there is insufficient experimental evidence available to judge their general validity.

Research most closely related to the present dy is n the area of analysis of errors and their causes in large sof systems. [Endres 75] discusses and categorises errors and error quencies during the internal testing phase of the IBM DOS/VS system. In [Thayer 78] data collected from four large software development projects is analysed. [Hamilton 78] applies the well known execution time model [Musa 80] to measure the operational reliability of computer center software, and [Glass 80] examines the occurrence of persistent bugs and their causes in operational software. Another useful study is [Maxwell 78], which tabulates and examines error statistics on software.

None of these studies tries to relate system reliability or the error frequencies to the usage environment of the software itself in a systematic manner. Results based on such measurements are essential in order to evaluate the system fault tolerance and automatic recovery features.

In an early study of failures at the SLAC (Stanford Linear Accelerator Center) computation facility, [Butner 80] and [Iyer 82a] found a strong correlation between the occurrence of failures and the level of

system activity at the time of failure. A more detailed and accurate analysis of failures on a VM/370 system (in service at SLAC since February 1981) confirmed this relationship [Rossetti 82]. In addition this study found that a significant proportion (16 percent) of software-related system failures were due to hardware problems. In many of these cases it was determined that the system should have been designed to continue operation at least in a degraded mode. To the authors' knowledge there are no other experimental studies reported in the literature on hardware-software interaction.

More recently [Velardi 83] analysed the error recovery facilities on the MVS system. Data on error recovery showed that the system fault tolerance almost doubles when recovery routines are provided for failing programs, in comparison with the case where only system provided recovery management is available. The recovery routines are most effective in handling storage management problems (an important feature of MVS). However, even when recovery routines are provided, there is almost a 50% chance of system failure when critical system jobs are involved. Thus there is still considerable scope for improvement. Deadlocks, I/O and data management, and exceptions are the main problem areas.

Finally, a preliminary examination of the data appeared to indicate that the error detection in MVS is not always able to diagnose software problems resulting from a hardware failure. It was clear that further analysis was necessary to fully understand this problem.

3. THE DATA BASE

The automatic detection of a software error in MVS can be through hardware or software facilities. Hardware detects conditions such as overflows, addressing or divide exceptions and, is generally used to protect storage or other system resources from unauthorised access. Hardware detection manifests itself as a program interruption (program check). Software detects more complex conditions such as an incorrect parameter specification in a macro or the unvalid use of control statements. Data on the type of detection (hardware or software) and recovery is logged by the system on to a data set called SYS1.LOGREC. A description of error detection and recovery processing in MVS appears in Appendix A and, in [IBM 79].

Initially, the SYS1.LOGREC data set (which is in hexadecimal code), was compacted in order to extract the relevant information, and to provide explanations for hexadecimal codes. Then, the records believed to be repeated occurrences of the same problem were clustered. The number of observations in a cluster (SWPOINTS, HWPOINTS) and time span of the cluster (SWSPAN, HWSPAN) were also added to the record. The result of this manipulation was a data set ready for statistical analysis. The building of this data base is discussed in detail in [Velardi 83].

3.1 PROCESSING THE ERROR DATA

The raw LOGREC data includes CPU, channel, and device errors for all equipment in the installation. Initially the software records on the two IBM 3081's were selected for this analysis. In each software record there are a number of bits describing the type of error, its severity,

and the result of hardware and software attempts to recover from the problem. The general software error status indicators provided by the hardware and software are TYPE (of detection), EVENT (causing the detection) and ERRCODE (code or symptom of the error).[1] For the purposes of this study two additional data sets which contained information on hardware/software interaction were also generated:

1. Software errors found related to temporary hardware errors (HW/SW-Temp.).
2. Software failures found related to permanent hardware failures (HW/SW-Perm.).

The HW/SW-Perm. data set was created by matching the software records with the log (UNILOG) of all hardware failures manually maintained at CIT. The matched records were then inspected to confirm that the resulting data (nearly 70 failures) did indeed correspond to hardware-related software failures. The HW/SW-Temp. data set was obtained by matching the software errors with temporary channel and disk problems. The data on channel problems came from the <u>Channel</u> <u>Check</u> (<u>CCH</u>) records and from <u>Missing</u> <u>Interruption</u> <u>Handling</u> (<u>MIH</u>) records. The data on disk errors came from the system outboard records (OBR). Again the merged data set was carefully inspected to confirm that the records reasonably well corresponded to hardware-related software errors. Table 1 provides brief descriptions of the sources of data employed in this study (see [IBM 79] and [Butner 80] for a detailed description of these records). A sample of the hardware-related software records is given in Fig. 1. A summary of the data appears in Table 2. Interesting frequency plots of the data are given in Appendix B.

[1] The IBM names for these fields are [IBM 79]: TYPE - HDRTYP; EVENT - SDWERRA; ERRCODE - SDWACMPC.

TABLE 1

Sources of data

Type of Record	Explanation
Channel Check Record (CCH)	These records are generated for every channel error (includes Channel Control Checks, Channel Data Checks and Interface Control Checks). CCH's are temporary hardware errors and do not result in system termination.
Missing Interruption Handling (MIH)	MIH records are due to missing or pending device and channel end interruptions.
Out Board Records (OBR)	OBR records are generated for a wide range of events (normal and abnormal). The category used in this analysis is temporary and permanent device errors.
Software Records	Software records are generated for selected software events. Examples are invalid SVC, program checks, system abends or user abends which request a recording.
UNILOG	UNILOG is an installation maintained log of all software and hardware component and system failures.

HW/SW Perm.

OBS	TIMESTMP	SWPOINTS	SWSPAN	JOB	TYPE	EVENT	ERRCODE	RESULT	DEVICE
1	01MAR82:17:31:42	4	1	INIT	SWDSWERR	PROGABT	913000	SYSDAMAG	DRUM
2	05MAR82:11:02:33	8	0	PJG82700	SWDSWERR	ROUTSVC	0A000	SYSDAMAG	DRUM
3	05MAR82:11:04:50	4	0	PJG82700	SWDSWERR	ROUTSVC	0A000	SYSDAMAG	DRUM
4	05MAR82:11:04:51	6	0	PJG82700	SWDSWERR	PROGABT	106000	SYSDAMAG	DRUM
5	05MAR82:11:40:39	10	20	LOSTRECS	LOSTRECS	PROGCHK	000000	SYSDAMAG	DRUM
6	06MAR82:11:05:36	24	99	INIT	HWDSWERR	PROGCHK	0C4000	SYSDAMAG	CPU
7	10MAR82:13:43:59	46	658	LOSTRECS	LOSTRECS	PROGABT	000000	SYSDAMAG	DRUM
8	13MAR82:16:09:03	8	7	WYLBUR	SWDSWERR	SYSABT	13E000	SYSDAMAG	ELSE
9	13MAR82:18:09:04	458	2530	INIT	HWDSWERR	PROGCHK	0C4000	SYSDAMAG	ELSE
10	15MAR82:01:16:31	4	4	LOSTRECS	LOSTRECS	PROGABT	000000	SYSDAMAG	DISK
11	18MAR82:11:22:58	14	0	NONE-FRR	SWDSWERR	ROUTSVC	282000	SYSDAMAG	ELSE
12	22MAR82:08:59:42	4	16	JES2	HWDSWERR	PROGCHK	028000	SYSDAMAG	DRUM
13	22MAR82:08:54:59	4	40	JES2	SWDSWERR	ROUTSVC	C0D000	SYSDAMAG	DRUM
14	31MAR82:17:07:03	4	0	JGCCPY	HWDSWERR	PROGCHK	0C4000	SYSDAMAG	DRUM
15	31MAR82:17:11:27	6	14	RMF	SWDSWERR	SYSABT	0FE000	SYSDAMAG	DRUM
16	31MAR82:17:11:56	6	3	RMF	SWDSWERR	ROUTSVC	301000	SYSDAMAG	DRUM
17	14APR82:18:49:43	3	1044	MANY	SWDSWERR	ROUTSVC	05C000	SYSDAMAG	ELSE
18	26APR82:17:43:20	2	1	WYLBUR	SWDSWERR	SYSABT	222000	SYSDAMAG	DISK
19	27APR82:00:56:15	1	0	*MASTER*	OPERDERR	RESTART	071000	SYSDAMAG	DISK
20	27APR82:05:17:46	1	0	NBTRJCL	SWDSWERR	PROGABT	013000	SYSDAMAG	DISK

HW/SW Temp.

OBS	TIMESTMP	SWPOINTS	HWPOINTS	SWSPAN	HWSPAN	JOB	TYPE	EVENT	ERRCODE	RESULT	DEVICE
1	04MAR82:08:31:00	2	14	0	101	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	CCK
2	10MAR82:13:43:59	46	4	658	4	LOSTRECS	LOSTRECS	PROGAB	000000	N/A	CCK
3	15MAR82:01:16:31	4	746	4	607	LOSTRECS	LOSTRECS	PROGAB	000000	N/A	DISK
4	18MAR82:02:13:39	4	10	0	10	NONE-FRR	HWDSWERR	PROGCH	A00000	RETRY	CCK
5	18MAR82:02:13:55	2	2	0	0	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	DISK
6	18MAR82:11:27:21	8	2	179	0	JES2	SWDSWERR	ROUTSV	C0D000	RETRY	MIH
7	18MAR82:14:31:52	6	4	0	378	NONE-FRR	SWDSWERR	ROUTSV	282000	N/A	MIH
8	22MAR82:05:25:31	2	20	0	168	ORVYLRLG	SWDSWERR	SYSABT	222000	SYSDA	CCK
9	22MAR82:05:25:31	2	4	0	17	ORVYLRLG	SWDSWERR	SYSABT	222000	SYSDA	MIH
10	22MAR82:15:48:31	2	4	0	85	P14VE436	SWDSWERR	SYSABT	222000	JOBTE	MIH
11	25MAR82:08:02:25	4	14	0	6	INIT	HWDSWERR	PROGCH	0C4000	SYSDA	DISK
12	26MAR82:03:22:33	28	14	543	566	WYLBUR	SWDSWERR	SYSABT	222000	SYSDA	MIH
13	29MAR82:10:13:34	68	2	177	0	MANY	SWDSWERR	PROGAB	D23000	SYSDA	DISK
14	29MAR82:11:19:52	2	2	0	0	TAPE	SWDSWERR	SYSABT	222000	JOBTE	MIH
15	30MAR82:14:54:03	4	66	0	64	LOSTRECS	LOSTRECS	PAGERR	000000	N/A	CCK
16	31MAR82:17:07:03	4	26	0	964	JGCCPY	HWDSWERR	PROGCH	0C4000	RETRY	CCK
17	01APR82:13:38:09	1	1	0	0	JXH16390	HWDSWERR	PROGCH	0C4000	TASKT	CCK
18	04APR82:12:34:25	1	1	0	0	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	DISK
19	04APR82:14:05:44	2	1	698	0	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	DISK
20	08APR82:19:22:29	2	0	0	713	JES2	SWDSWERR	SYSABT	351000	SYSDA	DISK
21	08APR82:19:27:05	145	8	2621	713	MANY	HWDSWERR	PROGCH	0C1000	SYSDA	DISK
22	09APR82:05:35:33	1	3	0	189	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	MIH
23	10APR82:16:16:48	1	6	0	654	M92E1928	SWDSWERR	SYSABT	322000	TASKT	CCK
24	11APR82:14:08:12	1	5	0	0	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	CCK
25	14APR82:03:15:05	1	46	0	1888	MILTEN	SWDSWERR	ROUTSV	05C000	RETRY	MIH
26	28APR82:11:23:59	140	1	604	0	MANY	HWDSWERR	PROGCH	0C1000	SYSDA	DISK

Figure 1: Sample of hardware/software errors

It can be seen from the data in Fig. 1 that it is not unusual to have more than one software record for a permanent hardware problem (i.e. HW/SW-permanent). Obseravations 2-4 and 12-13 are some examples. For temporary hardware problems note that not only some of the observations are very close in time they also refer to different hardware or software problems. For example observations 4 and 5 indicate that two software errors occurred in connection with a channel check and a temporary disk error on different programs. The time vicinity of these errors suggests that the cause of these problems was common. It is clear that the system was not able to diagnose and relate these records (e.g. two SW records, one CCH and one OBR, for the temporary hardware problem). A detailed analysis of the data (both HW/SW-perm. and HW/SW-temp.) confirmed that the system is seldom able to diagnose a hardware related software error.

TABLE 2

Summary of the data

Period of Study: March 1982 - May 1983

Data Set	Source	Freq.
All SW Errors	SW Records	1547
All Permanent HW Failures	UNILOG	264
All Temporary HW Errors	CCH, OBR	4461
SW Errors Related to Temporary HW Errors	SW Records/ CCH, OBR	108
SW Errors Related to Permanent HW Failures	SW Records/ UNILOG	69

The next section investigate the detection of software errors. Particular attention is paid to the detection of software errors related to temporary and permanent hardware problems.

4. ANALYSIS OF ERROR DETECTION

This section investigates the the detection of software errors in MVS. In particular, the following points are considered:

1. The relationship between the type of software problem and the type of detection (i.e. hardware or software).
2. The impact of hardware or software detection on system recovery.
3. The detection of software errors found to be hardware-related.

4.1 ERROR CLASSIFICATION

In common with other analyses of this type, the ERRCODE provided by the system were grouped into classes of similar problems. The error classes were chosen to reflect commonly encountered problems. In addition, other studies of this nature were also consulted (e.g., [Thayer 78], [Endres 75], [Rossetti 82]). Finally, it was important to make sure that each error category had a statistically significant number of errors in it.

Seven classes of errors were defined:

1. Control: indicates the invalid use of control statements and invalid supervisor calls.
2. I/O and data management: indicates a problem occurred during I/O management or during the creation and processing of data sets.

3. Storage management: indicates an error in the storage allocation/de-allocation process or in virtual memory mapping.

4. Storage exceptions: indicates addressing of non-existent or inaccessible memory locations.

5. Programming exceptions: indicates a program error other than a storage exception.

6. Deadlocks: indicates a system or operator detected endless loop, endless wait state or violation of system or user defined time limits.

7. Lost Records: indicates that the error recording process was itself affected by an error.

4.2 ERROR DETECTION STATISTICS

There are significant differences in the error distributions between the two detection mechanism. Table 3 gives the percentage distribution of the errors during the analysed period. On the average, the two major error categories are storage exceptions (25%) and storage management (26%).

It can be seen that all exception type problems are detected by hardware and storage management type problems are detected by software. In the case of control and I/O problems, it is found that almost twice as many are software-detected. An analysis of the hardware-detected control and I/O problems showed that these were in fact forced program checks and were detected as a result of specific software traps. Note from Table 3 that storage related problems dominate both hardware and software-detected errors. Recall that a major feature of the MVS operating system

TABLE 3

Distribution of error categories

	Hardware Detected		Software Detected		All
Error type	Freq.	%	Freq.	%	%
Storage management	11	1.9	395	44.2	26.2
Storage exceptions	382	67.0	0	0.0	24.7
Deadlocks	0	0.0	310	34.6	20.2
I/O and data management	45	7.9	116	13.0	10.5
Programming exceptions	114	19.9	0	0.0	7.4
Control	18	3.2	50	5.6	4.4
Invalid	1	0.1	23	2.6	6.6
ALL	571	100.0	894	100.0	100.0

is the multiple virtual storage organisation. Storage management is a high volume activity and is critical to the proper operation of the system. One might therefore expect its contribution to errors to be significant.

4.3 ERROR DETECTION AND RECOVERY

In MVS the system can recover from an error by a retry or by aborting the job or task (a module of the job) in progress [IBM 80]. If the job or task is critical for system continuation, abortion will cause system failure. Table 4 provides information on how an error was handled. The table shows that a hardware-detected error is more likely to result in a system failure and less likely to be retried successfully than a software-detected error.

TABLE 4

Effectiveness of the recovery

Detection	Freq.	JOBTERM %	TASKTERM %	RETRY %	FAILURE %
Hardware	571	0.9	45.2	24.0	29.9
Software	894	20.0	26.6	35.6	17.7
All*	1547	13.0	33.5	31.1	22.4

* This includes Lost Records and Operator detected errors also.

Recovery routines are specified in MVS for major system functions [Auslander 82]. Table 5 relates the provision of recovery routines to the detection mechanisms. We find that recovery routines are specified for almost twice as many software-detected problems than for hardware-detected. The table shows that software-detected problems are better handled (higher chance of a recovery than for hardware-detected prob-

TABLE 5

Effect of recovery routines

Error Type	Recvy Routine Provided %	Failures (Rcvy Routine Provided) %	Failures (Rcvy Routine Not Provided) %
Hardware	43.5	27.6	31.6
Software	84.8	13.3	42.1
All	66.1	16.8	34.8

lems). In both cases however we find that the availability of a recovery routine substantially improves the recovery probability. An important reason for the better performance of software-detected problems, is due to the fact that software detects most (or all) management type problems. Since storage management is an important function of MVS and it is more carefully designed and better protected by recovery routines. Also the system has more information available regarding a software detected problem than one detected by the hardware.

4.4 DETECTION OF HARDWARE-RELATED SOFTWARE ERRORS

Our previous analysis [Velardi 83] appeared to indicate that the error detection mechanism on MVS is not always able to diagnose software problems resulting from a hardware failure. Recall that the error data set used in this study contains information on software errors and failures found to be related to both temporary and permanent hardware problems.

TABLE 6

HW/SW errors - detection

	HW/SW-Temporary		HW/SW-Permanent		All SW*	
Detection	Freq.	%	Freq.	%	Freq.	%
Hardware	23	21.3	27	39.1	521	38.0
Software	64	59.3	30	43.5	800	58.5
Lost record	21	19.4	10	14.5	46	3.4
Operator	0	0.0	2	2.9	1	0.1
Total	108	100.0	69	100.0	1368	100.0

* Note: This does not include hardware-related problems

Table 6 analyses the detection of software errors found to be hardware-related. It can be seen from Table 6 that lost records are a significant proportion of hardware-related software errors. Note also the fact that more than 40 percent of all lost records occur in combination with a HW/SW error (whereas HW/SW errors are only 11.0 percent of all software errors). This seems to show that software error data collection itself is affected by the occurrence of a hardware error. Further investigation of this problem revealed that the job name of the hardware record associated with the software error tagged "LOST" generally indicated a system critical job. In addition lost records commonly appear in very large clusters indicating the persistency of a problem and usually result in system termination. It appears from the data that

such an occurrence can almost always be considered as a symptom of a hardware-related software problem.

5. ANALYSIS OF HARDWARE RELATED SOFTWARE PROBLEMS

This section analyses temporary and permanent hardware-related software problems. Significant features of hardware-related software errors are determined and their effect on recovery management is examined.

TABLE 7

Device involvement statistics

Device	HW/SW-Temporary		HW/SW-Permanent		All HW/SW
	Freq.	% (All SW Errors)	Freq.	% (All SW Failures)	% (All SW Errors)
CPU/Channel	76	4.9	20	7.0	6.1
Disk	32	2.1	42	14.6	4.8
Other	0	0.0	7	2.4	0.1
Total	108	7.0	69	24.0	11.0

Table 7 shows the frequency and percentage of hardware devices involved in software errors. Disks or channels are almost always involved. CPU and channel are considered together because the IBM 3081 contains both in one box and usually a channel problem also effects the CPU. The table also shows that about 11 percent of software errors are found

related to a hardware problem. About 7 percent of all <u>software errors</u> were related to temporary hardware problems. Nearly 25 percent of all <u>software failures</u> however were related to permanent hardware problems. The statistics on permanent hardware failures is somewhat higher than the results on VM/370 reported in [Rossetti 82]. That study found 16 percent of all software failures were hardware-related.

Table 8 provides statistics on hardware related software errors, i.e. Time Between Errors, the number of records (SWPOINTS) in a cluster (i.e. referring to the same problem) and the time span (SWSPAN) of the error (time between the first and the last record in a cluster). It is noted that both HW/SW-Temp. and HW/SW-Perm. have larger clusters and larger error handling times (i.e. SWSPAN) in comparison with all SW errors. The permanent failures have the larger times of the two. It was also observed that several of the large clusters had many jobs involved.

Summarsing, we find that the key features of hardware-related software problems are that they are very likely to result in lost records, occur in large clusters and involve many jobs.

TABLE 8

Statistics on hardware/software interaction

TIME BETWEEN ERRORS (Hours)				
	HW/SW-Temp.	All SW Errors	HW/SW-Perm.	All SW Failures
Mean	100.9	7.9	159.4	44.8
Standard deviation	208.3	12.8	304.8	108.8
Median	26.2	2.5	43.5	6.3

SWSPAN (Seconds)				
	HW/SW-Temp.	All SW Errors	HW/SW-Perm.	All SW Failures
Mean	91.0	49.5	205.4	47.9
Standard deviation	312.4	203.8	958.6	183.7
Median	0.0	0.0	0.0	0.0

SWPOINTS				
	HW/SW-Temp.	All SW Errors	HW/SW-Perm.	All SW Failures
Mean	11.5	4.2	16.9	4.8
Standard deviation	33.1	15.7	60.3	23.3
Median	2.0	2.0	4.0	2.0

5.1 RECOVERY OF HARDWARE-RELATED SOFTWARE ERRORS

This section analyses the recovery mangement of temporary and permanent hardware-related software problems. Recall that in handling a software problem the system can recover by issuing a retry, or by aborting the current job or task (a module of the job) in progress. If the job involved is critical for system continuation, system failure will result.

TABLE 9

Specification of recovery routines for HW/SW errors

Error Type	Recvy Routine Provided %	Failures (Rcvy Routine Provided) %	Failures (Rcvy Routine Not Provided) %
Temporary	62.9	20.6	84.2
Permanent	46.4	100.0	100.0
All	56.5	46.0	92.0

Recovery routines are specified in MVS for major system functions. Table 9 shows that software errors related to permanent hardware failures have a lower probability of having recovery routines specified than software errors related to temporary hardware errors or normal software errors. The figure is almost a third lower. Comparing Tables 9 and 5, it is also clear that, although recovery routines are specified for almost the same proportion of HW/SW-temporary errors as for all software

errors, they are not nearly as effective. In addition, the percentage of failures when recovery routines are not provided is substantially higher. Thus, the system recovery management is significantly less effective in handling a HW/SW error than it is in dealing with a software problem in general. This is significant since it points to a particularly weak aspect of the system. It may be argued that a better provision of recovery routines specifically geared toward the hardware-software interaction could considerably alleviate the problem.

TABLE 10

HW/SW-Temporary: Recovery management

Error type	TOTAL Freq.	TOTAL %	CCH %	MIH %	DISK %
Retry	25	23.2	20.5	18.9	31.3
Task Term.	16	14.8	10.3	13.5	21.8
Job Term.	19	17.6	7.7	43.2	0.0
Failure	25	23.2	18.0	16.2	37.5
Lost Records	23	21.3	43.6	8.1	9.4
All	108	100.0	36.1	34.3	29.6

Tables 10 and Table 11 provide information on recovery from HW/SW-Temporary errors. It can be seen from the table 10 that MIH (Missing Interruption Handling) causes the highest job and task terminations and sytem damage. These are seen from table 11 to be most closely related to

TABLE 11

HW/SW-Temporary: Error types

Error type	TOTAL Freq.	%	CCH %	DISK %	MIH %
Control	4	3.7	0.0	3.1	8.1
Deadlocks	29	26.9	15.4	0.0	62.2
I/O and data management	7	6.5	2.6	9.4	8.1
Storage management	23	21.3	18.0	31.3	16.2
Storage exceptions	12	11.1	15.4	18.8	0.0
Programming exceptions	8	7.4	0.0	21.9	2.7
Unclassified	25	23.2	48.7	15.6	2.7
All	108	100.0	36.1	29.6	34.3

deadlocks. This is quite reasonable since MIH are due to interrupts which are not completed in a specified time. The deadlocks are most commonly due to the detection of a wait state or an endless loop. Retries are also the lowest for MIH since most of them are deadlocks. More than 40% of the channel related software errors result in a lost record. We find that in most of these cases both the hardware and the software problem have large clusters associated with them. The disk and channel errors most commonly manifest themselves as storage problems or exceptions. This could also imply that the real problem was not in the channel but perhaps in main storage which resulted in both the channel error and the software record.

It is significant to note that 23% of HW/SW-temporary errors result in system failure. Taking this and HW/SW-permanent failures into account, it was found that nearly 35 percent of all software failures are hardware related. In addition, it was found that most of the lost records also resulted in system termination. Thus the true percentage of software failures (in our data) which are hardware related, is over 40 percent.

In summary, the analysis shows that recovery management of HW/SW errors, is significantly less effective than that of software errors in general. In many of these cases it was felt that the system could have been designed to continue in a degraded mode. At least the software should be capable of recognising a hardware failure and take the offending component off-line or put the system in a wait state.[2]

Software problems related to temporary hardware errors are not well managed either. The system has a low fault tolerance for these errors. Over 40 percent of all software failures are hardware related. It is believed that an important reason for this is the inadequate communication between the hardware and software regarding the occurrence of errors. If a hardware error was diagnosed and tagged as a potential software error, it is possible that better recovery could be designed. This would be especially true if the system was geared to recognise certain patterns in these errors (such as those observed here) and classify them as potential software problems. More data analysis and experimentation is necessary before this can be achieved in a reliable manner.

---

[2] Although this capability does exist in MVS in handling some hardware problems e.g. channel errors, there is no specific provision for handling HW/SW errors in general.

6. CONCLUSIONS

It has been the purpose of this paper to analyse the interaction between hardware and software as it relates to system reliability. It was seen that a hardware-detected error is more likely to result in a system failure than a software-detected problem. An important reason for the better performance of software-detected problems, is due to the fact that software detects most (or all) management type problems. This is an important function of MVS and hence more carefully designed and better protected by recovery routines.

Statistics on HW/SW errors shows that about 11 percent of software errors are hardware related. About 7 percent of software errors were related to temporary hardware problems; more than 24 percent of all software failures however were related to permanent hardware problems.[3] Taking all hardware errors into account (HW/SW-Temp. and HW/SW-Perm.) over 40 percent of software failures were determined to be hardware related.

Importantly, the analysis indicates that there is poor communication between facilities detecting hardware and software problems. An analysis of the data clearly shows that the system is not able to diagnose the fact that a software error may be hardware related. The key features of HW/SW errors identified in our data were:

1. Both hardware and software errors occur in large clusters
2. The HW/SW errors have a significant percentage of lost records.
3. The SW record in a HW/SW error may have many jobs involved.

---

[3] In comparison [Rossetti 82] found that 16 percent of software failures on VM/370 were hardware related.

4. The system recovery managment is less likely to recover from a HW/SW error than a software error in general.

It is suggested that some of the error patterns found in this study could form the basis for detection of hardware related software errors. It is of course possible the both the hardware error and the software error indicate no more than a symptom of the real problem. There is some evidence in our data to suggest that this is a possible scenario. However, if the detection was better coordinated, it is possible that at least system termination due to temporary hardware problems could be reduced. Better communication between the hardware and software error detection mechanisms may be an area where further effort toward alleviating this problem can be directed. There is no doubt that more data analysis and experimentation is necessary before patterns found in this study can be used as a basis for a suitable detection policy.

7. ACKNOWLEDGMENTS

The authors would like to thank Prof. E.J. McCluskey for his interest in this work and for extensive discussions during the period of this study. Special thanks are extended to Larry Rivers and Lincoln Ong at CIT and, to Dave Rossetti and Bill Weeks at SLAC for valuable discussions. Acknowledgments are also due to Dorothy Andrews for her careful reading of an earlier draft of this paper.

This work was supported in part by the U.S. Army Research Office under contract number DAAG29-82-K-0105, the Ugo Bordoni Foundation and the Italian National Research Council (CNR). The views, opinions, and/or findings contained in this document are those of the authors and

26

should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other official documentation.

REFERENCES

[Auslander 81] M.A. Auslander, D.C. Larkin and A.L. Scherr, "The evolution of the MVS operating system," IBM Journal of Research Development , Vol. 25, No. 5, September 1981.

[Butner 80] S.E. Butner and R.K. Iyer, "A statistical study of reliability and system load at SLAC," Digest, Tenth International Symposium on Fault Tolerant Computing, Kyoto, Japan, October 1980.

[Curtis 80] B. Curtis, "Measurement and experimentation in software engineering," Proceedings of the IEEE, Vol. 68, No. 9, pp. 1144-1157, September 1980.

[Endres 75] A. Endres, "An analysis of errors and their causes in systems programs," IEEE Trans. Software Engineering, Vol. SE-1, No. 2, pp. 140-149, June 1975.

[Glass 80] R.L. Glass, "Persistent software errors," IEEE Trans. Software Engineering, Vol. SE-7, No. 2, pp. 162-168, March 1981.

[Goel 80] A.K. Goel, "A summary of the discussion on 'An analysis of competing software reliability models'," IEEE Trans. Software Engineering, Vol. SE-6, No. 5, pp. 501-502, September 1980.

[Hamilton 78] P.A. Hamilton and J.D. Musa, "Measuring reliability of computation center software," Proc. Third Int. Conf. Software Engineering, Atlanta Georgia, pp. 29-36, May 1978.

[Hecht 80a] H. Hecht, "Current issues in fault tolerent software," Proceedings COMPSAC 80, Chicago Illinois, pp. 603-607, November 1980.

[Hecht 80b] H. Hecht, "Mini-tutorial on software reliability," Proceedings COMPSAC 80, Chicago Illinois, pp. 383-385, November 1980.

[IBM 81] IBM Corp., OS/VS2 MVS, System Programming Library: MVS Diagnostics Techniques, Order No. GC28-0725, 1981.

[IBM 80] IBM Corp., OS/VS2 MVS, System Programming Library: Supervisor, Order No. GC28-1046, 1980.

[IBM 79] IBM Corp., OS/VS2 MVS, System Programming Library: SYS1.LOGREC Error Recording, Order No. GC28-0677-5, 1979.

[Iyer 82a] R. K. Iyer, S. E. Butner, and E. J. McCluskey, "A statistical failure/load relationship; Results of a multi-computer study," IEEE Transactions on Computers, July 1982.

[Iyer 82b] R.K. Iyer and D.J. Rossetti, "A statistical load dependency model for CPU errors at SLAC," The Dig. FTCS-12, Twelfth International Symposium on Fault Tolerant Computing, Santa Monica, California, June 1982.

[Maxwell 78] F.D. Maxwell, The Determination of Measures of Software Reliability, Final Report, NASA-CR-158960, The Aerospace Corporation, El Segundo California, December 1978.

[Melliar-Smith 81] P.M. Melliar-Smith and R.L. Schwartz, "Current progress on the proof of SIFT," The Dig. FTCS-11, Eleventh International Symposium on Fault Tolerant Computing, Portland, Maine, June 1981.

[Musa 80] J. Musa, "The measurement and management of software reliability," Proc. IEEE, Vol. 68, pp. 1131-1143, September 1980.

[Rossetti 82] D.J. Rossetti and R.K. Iyer, "Software related failures on the IBM 3081: A relationship with system utilization," Proc. COMPSAC 82, Chicago, Illinois, November 82.

[Thayer 78] T.A. Thayer, M. Lipow and E.C. Nelson, Software Reliability: Study of Large Project Reality, TRW Series of Software Technology, Vol. 2, North-Holland, 1978.

[Velardi 83] P. Velardi and R.K. Iyer, A Study of Software Failures and Recovery in MVS, CRC Tech. Report No. 83-7, Center for Reliable Computing, Stanford University, August 1983.

Appendix A

MVS ERROR DETECTION AND RECOVERY PROCESSING

A.1 ERROR DETECTION

The supervisor in MVS offers many services to detect and process abnormal conditions during system execution.

1. The hardware detects conditions such as memory violations, program errors (arithmetic exceptions, invalid operation codes), addressing errors and password checking on critical system resources.

2. The software also provides detection of software problems.

   The data management and supervisor routines ensure that valid data are processed and non-conflicting requests are made. Examples are the incorrect specification of a parameter in a control structure or in a system macro, or a supervisor call issued by an unauthorized program.

   The installation might improve the system error detection capability by means of a software facility called Resource Access Control Facility (RACF). The RACF is used to build detailed 'profiles' of system software modules. These profiles are defined in order to inspect the correct usage of system resources.

   The user can also employ other software facilities to detect the occurrences of selected events. "Appendages" are routines

that enable the user to get control during different phases of an I/O operation. The "Servicability Level Indication Processing (SLIP) aids in error-detection and diagnosis also. The SLIP command allows the user to traps that cause a program interruption when particular events are intercepted. The user might also define his own detection mechanisms by means of the "Set Program Interruption Element" (SPIE) macro. This macro instruction detects programmer defined exceptions like using an incorrect address or attempting to execute privileged instructions. Using these facilities, user defined error conditions can be detected in addition to system provided program checks.

3. The operator might detect some evident error condition and decide to cancel or restart the job. For example, the operator can detect loop conditions or endless wait states.

A.2 RECOVERY PROCESSING

Whenever a program is abnormally interrupted due to the detection of an error, the Supervisor gets control. If the problem is such that a further processing could degrade the system or destroy data, the Supervisor gives control to the Recovery Termination Manager (RTM). If a recovery routine is available for the problem program, RTM gives control to this routine before processing the program termination.

Recovery is designed as a means by which the system can prevent total loss. The purpose of a recovery routine is to free the resouces kept by the failing program (if any), to locate the error and to request either for a continuation of the termination process or for a retry. Recovery

routines are generally provided to cover all MVS functions [Auslander 81]. It is however the responsibility of the installation or of the user to write recovery routine for other programs.

More than one recovery routine can be specified for the same program; if the latest recovery routine asks for a termination of the program, the RTM can give control to another recovery routine (if provided). This process is called 'percolation'.

The percolation process ends if either a routine issues a valid retry request, or no more routines are available. In the latter case, the program and its related subtasks are terminated. The termination of a program might imply the termination of jobstep. If a valid retry is requested, a retry routine restores a valid status, using the information supplied by the recovery routine(s), and can give control to the program. In order for a retry to be valid the system should verify that there is no risk of recurrence of the error to the same recovery routine, and that the retry address is properly specified. Figure 2 illustrates the steps in the recovery process.

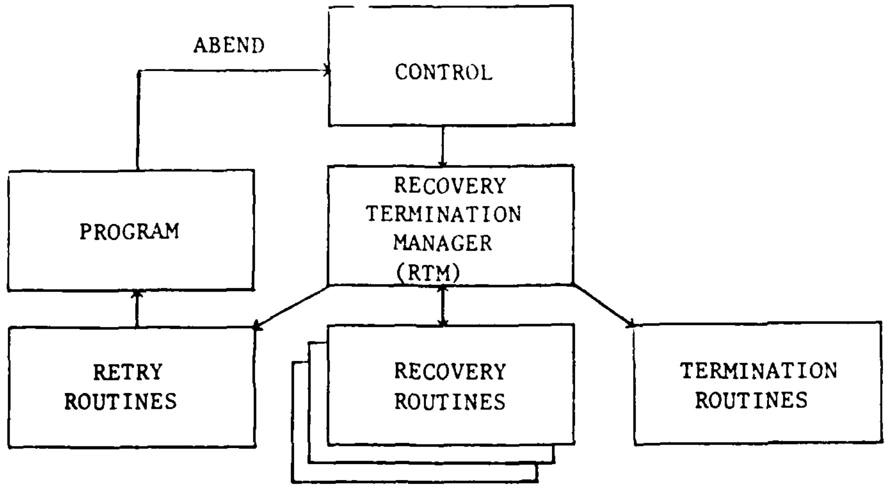


Figure 2: Software handling of software errors on MVS

A.3 ERROR RECORDING ON SYS1.LOGREC

Before a recovery routine takes control, the RTM inititialises a work area called the System Diagnostic Work Area (SDWA). This area is by the RTM to communicate with the recovery routines and, to log information regarding the error. Thus at the end of the recovery process the SDWA contains a history of the incident and the associated recovery process. At the end of the recovery process the RTM invokes the error recording routines to generate a record of the incident. The data set containing this information is called SYS1.LOGREC.

A software record also contains the information about the event (EVENT) that caused the record to be generated, and a 12 bit error symptom code (ERRCODE) describing the reason for the program abnormal termination. These codes are issued by the system or by the problem program

that used an ABEND macro instruction. The system and user completion codes appear together in the ERRCODE field. User codes are meaningful only for specific applications.

Table 12 describes the values assumed by the variable EVENT. Table 13 gives some example of common system ERRCODE's encountered in this study. The detection mechanism and the action taken by the system are also described. More than 500 different ERRCODE's are issued by the system for a problem program.

Traces of the recovery process are recorded on LOGREC. This includes the name and the type of the recovery routine which handled the problem (RECNAME), the result (RESULT) of the recovery process and the impact of the error on the related jobstep (JOBTERM). A description of these fields is given in Table 14. Other data collected during the recovery process, includes detailed program status information such as the contents of registers and the program address space identifier. This can be helpful in error diagnosis.

During the recovery process the system basically attempts to maintain operation despite an error. It is possible that the recovery process itself encounters the same error. In this case, there exists the risk of recursive recovery processes, or the generation of bad data. However, such occurences can be detected by analyzing the SDWA field into LOGREC. If the jobname for example is 'NONE-FRR', this indicates that the record is generated by a functional recovery routine during a recovery attempt. Finally, if the recording process was also affected by an error, a LOSTREC value appears in the TYPE field.

TABLE 12

Event that caused program termination

Variable EVENT Values	Meaning
MACHECK	A hardware event caused a machine check that could not handle the problem
PROGCHECK	A program check interrupt occurred due to the detection of some exception or to the violation of some memory protection mechanism
TRSFAIL	A translation error, e.g., an error occurred during the storage allocation process
RESTART	The operator pressed the restart key
ROUTABT	A system service routine detected an invalid SVC and issued an abnormal termination of the program (ABEND)
ROUTSVC	A system routine issued an invalid supervisor call (SVC)
PROGABT	The program itself requested the ABEND
SYSABT	The system detected a problem and forced a program ABEND

TABLE 13

Examples of ABEND reason codes

Hex code	Explanation	System action
05A	A service routine that handles real storage deallocation received an invalid address	The program that called the service routine or the routine abnormally terminates
071	The operator determined that the program was in a loop or endless wait state	The operator pressed the RESTART key
0C1	Operation exception: an operation code is not assigned	A program interruption occurred; the task is terminated if no routine had been specified to handle the interruption
020	The error occurred during the creation of a data set due to the incorrect specification of some data parameter	The task is terminated if no routine has been specified for the problem program

TABLE 14

Recovery information

Variable name	Values	Meaning
RECNAME	8 character name	Name of the recovery routine which handled the problem
RESULT	RETRY	The recovery routine decide that a retry might be successful
	CONTTERM	The recovery routine asks to continue with termination (this might imply percolation)
JOBTERM	YES/NO	If JOBTERM=YES the entire jobstep has to be terminated

An error may have four possible effects:

1. RETRY: The system successfully recovered and returned control to the problem program.
2. TASK TERMINATION: The progrma and its related subtasks are terminated, but the system is not affected.
3. JOB TERMINATION: The job in control at the time of the error is aborted.
4. SYSTEM DAMAGE: The job or task in control at the time of the error was critical for system continuation. Thus job/task termination resulted in system failure.

Appendix B

SOFTWARE ERRORS - FREQUENCY PLOTS

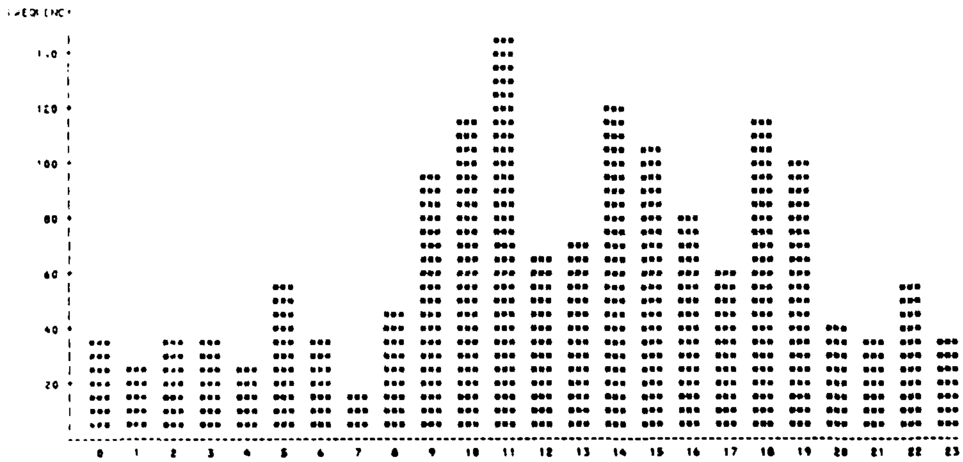


Figure 3: Hour of day plot of software errors

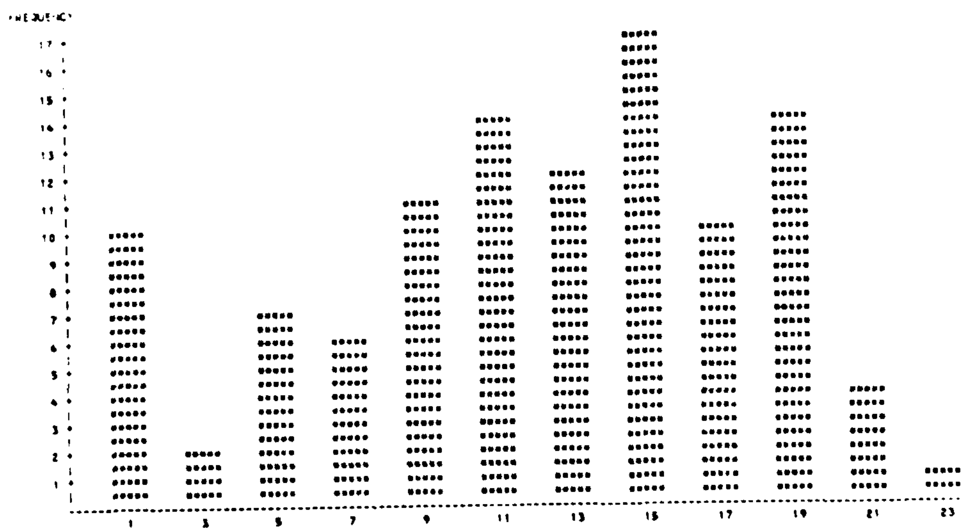


Figure 4: Hour of day plot HW/SW Temporary errors

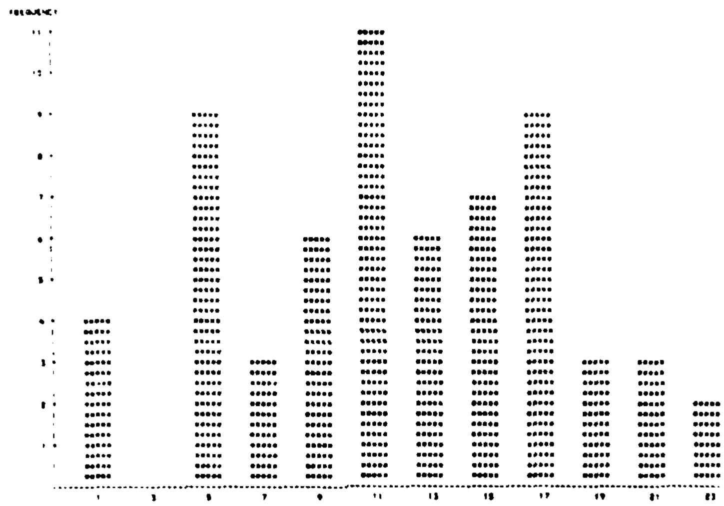


Figure 5: Hour of day plot HW/SW Permanent errors

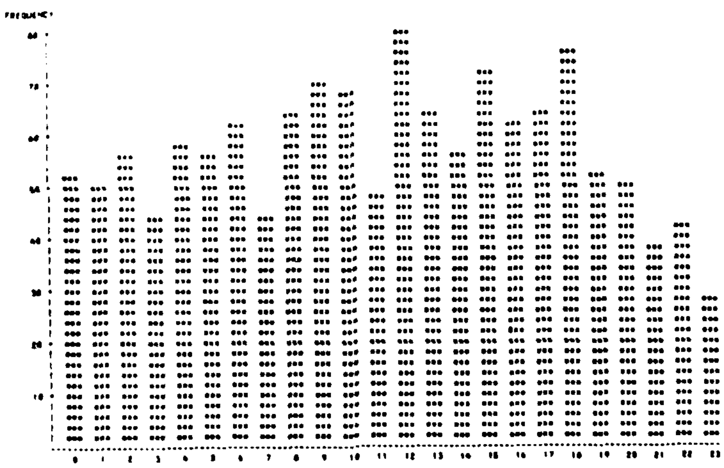


Figure 6: Hour of day plot all Temporary HW errors

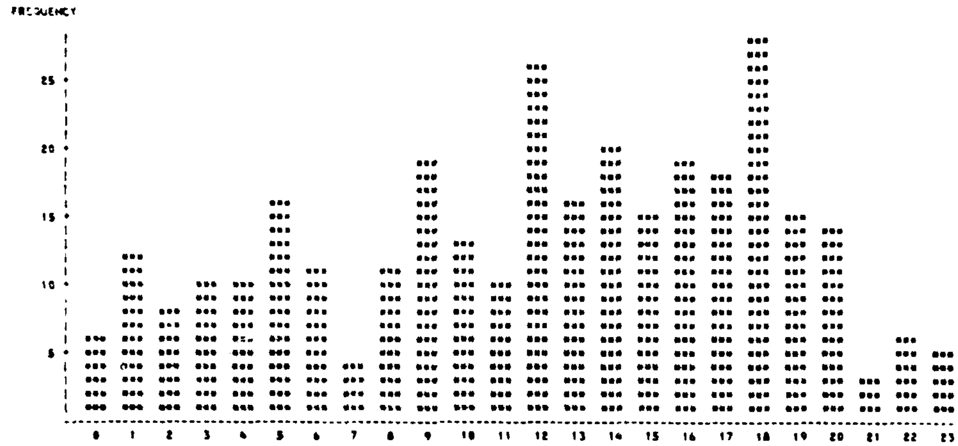


Figure 7: Hour of day plot all Permanent HW errors

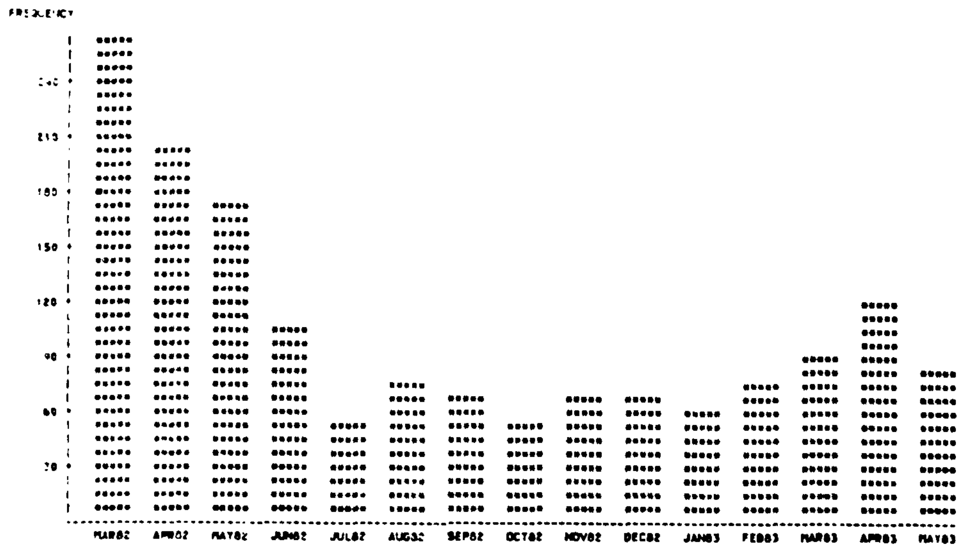


Figure 8: Frequency plot of all software errors by month